

Rapport de Projet
TaZmaniaTeaM

MAILLARD Alain (mailla_l)
GUIRAUD Camille (guirau_c)
LEVRAULT Hélène (levrau_h)
GISLAIS Sébastien (gislai_s)

20 juin 2008

Table des matières

1 Introduction

Bienvenue dans l'aventure des Epitoons ! Au fil de cette année et des différentes soutenances, vous avez pu suivre l'évolution de notre jeu au fur et à mesure de sa création et réalisation. Il est maintenant temps pour nous de vous montrer notre jeu fini, et prêt à jouer pour votre plus grand plaisir ! Nous vous présentons notre rapport de jeu, celui-ci vous fera découvrir notre tout premier projet en tant qu'élève de l'Epita.

Pour que vous puissiez tout d'abord connaitre notre équipe de réalisation, il ya aura en premier lieu sa présentation, l'histoire de sa formation et une présentation rapide de chacun de ces membres, suivi d'un récapitulatif de la répartition des tâches et de l'évolution de chacune d'entre elles et pour finir, d'un rapide rappel sur nos précédentes soutenances.

Nous vous ferons ensuite découvrir le monde de Taz ainsi que son gameplay. Nous vous expliquerons donc les étapes du développement de ce jeu divisées en catégories essentielles au jeu. Nous finirons par exposer les moyens déployés pour ce projet ainsi que les différents coûts engendrés.

Toute l'equipe de la TaZmaniaTeam vous souhaite une bonne lecture et vous donne rendez-vous sur http ://epitoons.free.fr/.

2 La TaZmaniaTeaM

2.1 Création du groupe

Notre groupe s'est formé assez rapidement. En effet beaucoup de spé nous avait prévenu : il fallait s'y prendre à l'avance car l'échéance arrivait vite et se mettre avec des gens que l'on connaissait relativement (pour éviter qu'un membre quitte l'école ou ne travaille pas). Nous étions ensemble lors de la pré-rentrée en mathématiques fin-août et nous avions pris l'habitude de travailler ensemble le soir. L'idée de nous mettre alors ensemble pour créer ce projet nous est tout de suite venu et ne fut pas regrettée (eh oui nous sommes toujours 4 !).

Il fallait ensuite trouver quel genre de jeu correspondrait aux attentes de chacun. Nous nous sommes finalement arrêter sur un jeu de plate-forme. Nous avons choisi Taz pour incarner notre personnage principal à cause/ou grâce (au choix) à Alain, notre bien-aimé chef de projet. En effet, celui-ci depuis qu'il est tout petit (ça n'a pas vraiment changé !) est fan de cet animal émettant des grognements étranges.

2.2 La team

Après ce flash-back, vous allez pouvoir connaitre (quelle chance !) chaque membre de notre équipe.

2.2.1 Alain

En commençant le projet en début d'année, j'ai tout de suite pensé que le projet était une opportunité pour en apprendre plus sur le développement d'un jeu vidéo et je n'aurais jamais imaginé que toute l'élaboration d'un jeu ce faisait comme cela... Ce projet a été un très bon commencement pour apprendre à programmer bien qu'au tout début je ne savais pas tellement par où commencer... Je fus désigné chef de projet et j'ai complètement assumé cette tâche bien qu'au premier abord je puisse paraitre un peu timide mais je me rend compte aujourd'hui que cela m'a bien été utile. Etant quelqu'un de très ouvert à la base (bien que timide) ma relation avec les autres membres du groupe était bonne ainsi que la communication bien que parfois l'information avait plus de mal à passer. Nous avons rencontré beaucoup de difficultés mais ce premier projet restera une bonne expérience pour l'avenir.

2.2.2 Camille

Au début de l'année, je n'imaginais que quelques lignes de code pouvaient donner un jeu vidéo... Ce projet m'a permis de comprendre le fonctionnement d'un jeu vidéo, ainsi que d'apprendre comment coder... La conception de ce jeu n'a été évidente à aucun moment de l'année. En effet, nous avons rencontré des problèmes de différentes sortes, jusqu'au dernier jour de travail. Il y a d'abord les problèmes liés au travail de groupe, comment réussir à mettre tout le monde d'accord sur chaque élément du projet ? Nous avons finalement réussi, et ce projet m'a permis d'évoluer dans les relations humaines au sein d'un groupe de travail. En effet, le travail en groupe permet une grande entraide et un apprentissage plus rapide. Ensuite, nous avons eu les problèmes liés à la programmation. En effet, comme les autres membres du groupes je n'avais jamais codé avant mon arrivée à EPITA. Nous avons donc eu des difficultés à commencer ce projet, et pour chaque idée nous avions des recherches à faire pour comprendre comment réaliser ce que nous voulions. J'ai trouvé ce projet intéressant à réaliser, malgré les difficultés rencontrées.

2.2.3 Hélène

En arrivant à Epita, on nous a tout de suite dit que ce n'était pas grave de n'avoir jamais codé. Et bien pour ce projet, ça aurait aidé quand même un petit peu ! En effet, au début j'étais un peu alarmée : comment créer un jeu vidéo quand on n'y connait rien ? ! Finalement, avec l'aide de certaines personnes et d'Internet (Google est ton ami) j'ai avancé petit à petit. Le système de répartition par binômes nous a aussi beaucoup aidés. Cependant, il serait faux de dire qu'après ce début un peu difficile, les choses sont devenues faciles. Ce projet nous a donné du fil à retordre sur différentes parties. Et c'est vraiment encourageant lorsque finalement les efforts et le travail payent ! Ce fut très intéressant de découvrir la réalisation d'un jeu et sa complexité. Cela m'a vraiment permis de me représenter réellement ce que je réalisai. D'autre part, la conception de ce projet en groupe m'a permis de pouvoir autant me faire aider qu'aider mes co-équipiers et c'est vraiment rassurant d'être entourée !

2.2.4 Sébastien

Le projet arrive à sa fin ! Pour moi, ce projet a été un projet de longue haleine. En effet, comme les autres membres du groupe, je n'avais jamais codé un projet d'aussi grande envergure. J'ai dû m'accrocher pour pouvoir réussir et participer du mieux que je pouvais. Au final, je suis plutôt content de ce que j'ai réalisé, de ce que nous avons réalisé. Ce que je retiendrai le plus c'est le travail de groupe et l'aventure humaine au cours d'une année qui nous réunit autour d'un projet dans le domaine qui nous passionne : l'informatique. Le travail de groupe a été très efficace et la communication entre nous a été primordiale. Comme nous étions tous débutants, personne n'a plus été avantagé ou désavantagé. Notre efficacité est liée au fait que nous nous sommes répartis en groupes de deux par domaine. De là, nous avons pu nous donner chacun de petits objectifs. C'est grâce à ça que j'ai pu tirer mon épingle du jeu en pouvant aider sur de petites choses et en pouvant aussi être aidé en retour. Même si notre projet n'est pas le meilleur du monde, les difficultés rencontrés ont été surmontées grâce à l'entraide et à la franche amitié qui nous a lié tout au long du projet.

2.3 Planning et répartition des tâches

2.3.1 Répartition des tâches

Tâches	Alain M.	Camille G.	Hélène L.	Sébastien G.
Moteur Graphique	⋆		⋆	
Moteur Physique	⋆			⋆
IA		⋆	⋆	
Audio	⋆	⋆	⋆	⋆
Réseau		⋆		⋆
Site Web	⋆	⋆	⋆	⋆

2.3.2 1re Soutenance

Tâches	Avancement
Moteur Graphique	Ébauche
Moteur Physique	Étudié
IA	Étudié
Audio	—
Réseau	—
Site Web	Consultable

2.3.3 2e Soutenance

Tâches	Avancement
Moteur Graphique	Bien avancé
Moteur Physique	Bien avancé
IA	Étudié
Audio	—
Réseau	Avancé
Site Web	À jour

2.3.4 3e Soutenance

Tâches	Avancement
Moteur Graphique	Terminé
Moteur Physique	Terminé
IA	Bien avancé
Audio	Ébauche
Réseau	Terminé
Site Web	À jour

2.3.5 Soutenance Finale

Tâches	Avancement
Moteur Graphique	Terminé
Moteur Physique	Terminé
IA	Terminé
Audio	Terminé
Réseau	Terminé
Site Web	Terminé

2.4 Présentation du projet

Il y a bien longtemps, dans une galaxie lointaine, très lointaine...

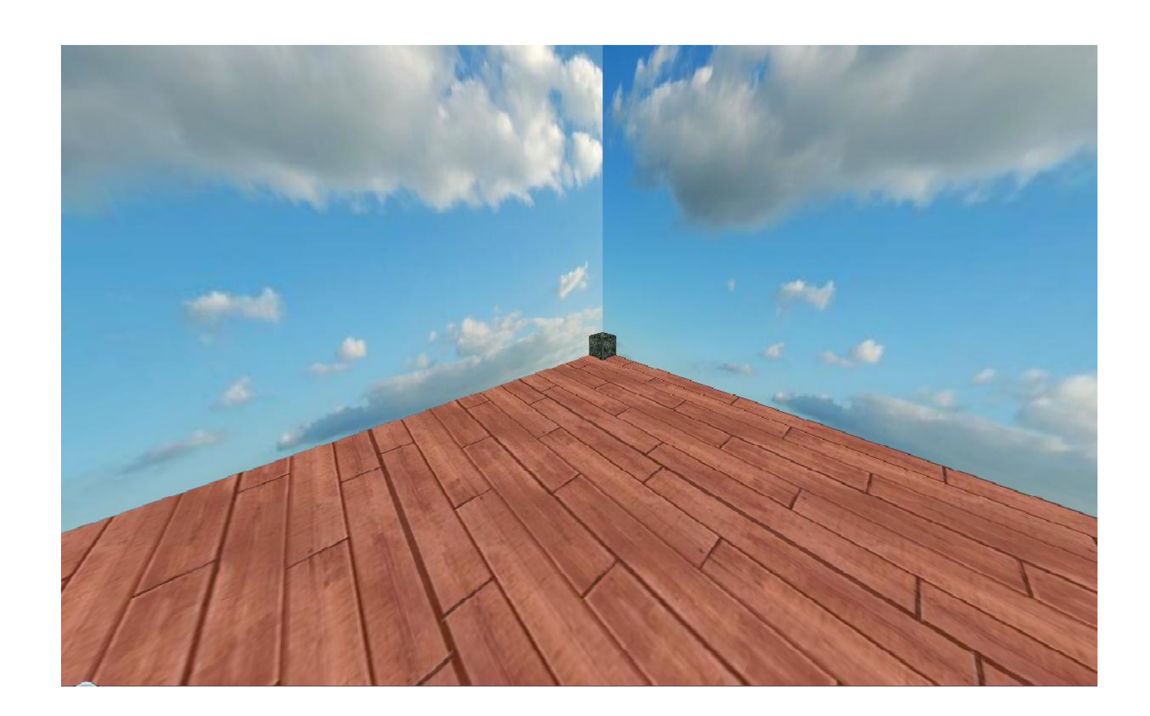
Quatre mille ans avant
L'empire Ionisien,
Epita est au bord du
Chaos. Dark Junior, le
Dernier apprenti du
Seigneur Noir Krisboul, a
Lancé l'invincible armada
Des Bocaliens à l'assaut de Ionis.
Ecrasant toute résistance les
Forces de Junior ont
Semé le trouble dans
L'ordre des Toons. De nombreux
Chevaliers sont tombés au
Combat, d'autres ont rejoint
Le rang des Bocaliens.
La planète Epita dans la
Bordure extérieure, sera
Bientôt le théâtre d'une
Terrible bataille entre
L'armée des Toons et les alliés de
Dark Junior. De l'issue du
Combat dépendra l'avenir
De Ionis...[1]

[1]Structure du texte tirée du jeu « Knights of the Old Republic »

2.5 Rappel des soutenances

2.5.1 1re Soutenance

Lors de notre première soutenance, nous avions présenté un cube qui se déplaçait dans toutes les directions dans une skybox. Nous avions aussi appliqué des textures sur notre cube ainsi que sur la skybox.



2.5.2 2e Soutenance

Notre jeu a vraiment évolué pour cette soutenance. Tout d'abord, nous avons commencé à créer le menu du jeu qui nous a permis de relier les différentes parties du projet (réseau, mini jeux...). En ce qui concerne le jeu en lui-même, nous avons créé des obstacles, et le moteur physique ayant évolué, Taz ne pouvait pas les traverser ! Taz n'était d'ailleurs toujours pas en 3D, mais nous avions appliqué des textures différentes sur les faces du cube : une image de Taz pour sa tête... De plus, un mini jeu était créé et terminé : Sam vs Bugs. Enfin, nous avions aussi créé le début du réseau. Nous pouvions alors communiquer avec un autre joueur.



2.5.3 3e Soutenance

A la troisième soutenance, nous n'avons toujours pas de personnage en 3D car nous avons vraiment eu de grosses difficultés avec le loader 3D (cf. partie moteur graphique). Mais le menu avait évolué ! Nous pouvons désormais accéder directement au... deuxième niveau ! Celui-ci a été créé depuis la soutenance 2. De plus, nous avons commencé le deuxième mini jeu. Enfin, le moteur physique a évolué et Taz peut sauter et évoluer sur les obstacles. Nous avions aussi commencé l'intelligence artificielle mais suite à de nombreuses difficultés (cf. partie intelligence artificielle) nous n'avions rien affiché à l'écran.



3 La création du jeu

3.1 Le Gameplay

Taz se retrouve tout d'abord coincé dans une forêt. Le seul moyen pour lui de sortir de celle-ci est d'atteindre l'autre côté de ce monde et de réussir le mini-jeu auquel il va accéder. A chaque fois qu'un mini-jeu est gagné, Taz se retrouve dans un autre monde. Dans le dernier monde, Taz doit battre le seigneur noir Krisboul pour finir le jeu. Pour cela Taz peut se déplacer à gauche, à droite, ainsi qu'en diagonale. Il peut avancer comme reculer et également sauter.

3.2 Les mondes, les objets

Taz se déplace donc dans différents mondes. Il doit éviter les obstacles tels que des caisses, mais également les monstres qui l'attaqueront lorsque celui-ci sera dans leurs champs de vision ! Taz devra éviter ces monstres car lorsqu'il se fait toucher par l'un d'eux il doit recommencer le monde dans lequel il est !

3.3 Le Moteur Graphique : Alain et Hélène

3.3.1 Prévisions

Pour la création d'un jeu vidéo, le moteur graphique est évidemment indispensable. Deux choix se sont donc offerts à nous quant à la librairie graphique à utiliser : OpenGL ou DirectX ? Evidemment lorsque l'on a commencé à poser quelques questions à ce sujet, ça a vite tourné en débats interminables. Finalement en voyant les tutoriaux disponibles pour OpenGL sur internet nous avons opté pour celui-ci, d'autant plus qu'une grande communauté de développeurs pouvait nous aider sur des forums spécialisés.

Voici ce que notre moteur graphique devait pouvoir faire à la soutenance :

- afficher une skybox
- afficher différents objets 3D (obstacles, ennemis…)
- une interface et un menu permettant de choisir son monde ou directement jouer à un mini-jeu.

3.3.2 Initialisation d'OpenGL (GLFW)

L'initialisation d'OpenGL… Le b-a-ba de tout projet commence par la création d'une fenêtre dans laquelle va se dérouler notre jeu. L'utilisation de la librairie glfw étant là pour nous faciliter ce travail, nous en avons profité pour créer cette fenêtre (glfwOpenWindow()). Une fois chose faite c'est beau nous avons une fenêtre qui s'affiche cependant on ne voit que du noir ! Pas si facile de s'y retrouver ! Le but, étant au final d'avoir quelque chose à afficher autre qu'un écran noir (c'est mieux !), nous nous sommes lancés dans l'affichage d'un cube, donner différentes coordonnées, OpenGL les reliera pour créer le dit cube, puis des déplacements de ce cube grâce à deux fonctions de la librairie glfw qui ne sont autres que gltranslate et glrotate. Après plusieurs essais, nous finîmes par obtenir un cube en 3D qui pouvait se déplacer et tourner en fonctions des touches du clavier que l'on avait assigné auparavant. Suite à cette magnifique réussite, nous avons tout de suite enchaîné avec la création de la map dans laquelle notre splendide cube allait se déplacer…

3.3.3 Les textures, la skybox, les obstacles et la caméra

Un cube qui se déplace c'est bien beau mais ça ne fait pas grand-chose et ce n'est surtout pas très joli c'est pourquoi nous lui avons appliqué des textures à partir de la librairie DevIL (Developer's Image Library). Il y en a d'autres mais c'est surtout celle qui est la plus adaptée à OpenGL et qui est très connue. Après, tout ce qu'il nous reste à faire c'est d'appliquer la texture voulue aux coordonnées que l'on veut et comme par magie la couleur de notre cube ou du décor est remplacée par la texture 2D que l'on avait choisie ! Grâce à cette application de textures, fini les couleurs toutes moches par défaut et bonjour à l'application d'images sur les différentes faces de notre cube et de notre skybox, qui n'est ni plus ni moins qu'un deuxième cube de taille plus

grande et qui englobe la scène, c'est ensuite la caméra qu'il faut bien placer pour avoir une vue d'ensemble correcte et non déformée.

Concernant les obstacles, le cube qui se déplace, n'est autre que notre personnage principal, et lorsqu'il deviendra grand, devra se battre contre des méchants pas beaux qui veulent dominer le monde ! Rien que ça ! C'est pourquoi nous devons créer des obstacles, ainsi que des monstres qui rendront encore un peu plus difficile le périple de notre Taz adoré. Pour se faire une idée, nous avons dessiné des cubes à différents endroits de la carte de façon à voir à peu près où ceux-ci seront positionnés par la suite, en attendant de mettre au point le loader 3D qui sera l'aboutissement de tout notre moteur graphique. Les obstacles sont représentés de la même manière que le cube principal c'est-à-dire que l'on défini les coordonnées de points formant des carrés où l'ont appliquait une couleur auparavant mais maintenant on applique une texture. On obtenait ainsi les obstacles que nous avions encore jusqu'à la troisième soutenance. Au niveau de la caméra nous avons opté pour l'utilisation d'une caméra mobile qui suit les mouvements de Taz. En effet en affectant à la caméra les mêmes variables que celles utilisées pour faire déplacer Taz permet aisément à la caméra de suivre notre personnage.

3.3.4 Le loader 3DS

Ah... la partie la plus problématique de tout le moteur graphique... Nous avons présenté le loader 3D à partir de la troisième soutenance dans une fenêtre séparée, chargeant un modèle en 3D avec la possibilité de zoomer sur celui-ci, de le faire tourner et même de le faire avancer. Cependant, il restait un très gros problème car, bien qu'il fonctionnait dans cette fenêtre séparée, le loader 3D nous jouait des tours dès qu'il s'agissait de charger un modèle 3D dans le jeu en lui-même.

Pour explication voici comment marche le loader 3D : il comprend deux parties distinctes :

- Premièrement le loader charge un fichier 3D de format .3DS dans notre jeu
- Ensuite il faut que le loader dessine le modèle dans la scène du jeu
- Enfin le loader applique les textures correspondantes sur les différents modèles chargés

Suite à tous ces chargements, le modèle s'affiche avec les textures correspondantes appliquées aux bons endroits formant ainsi un beau modèle 3D, que ce soit de notre Taz ou bien de nos monstres, décor, etc etc...

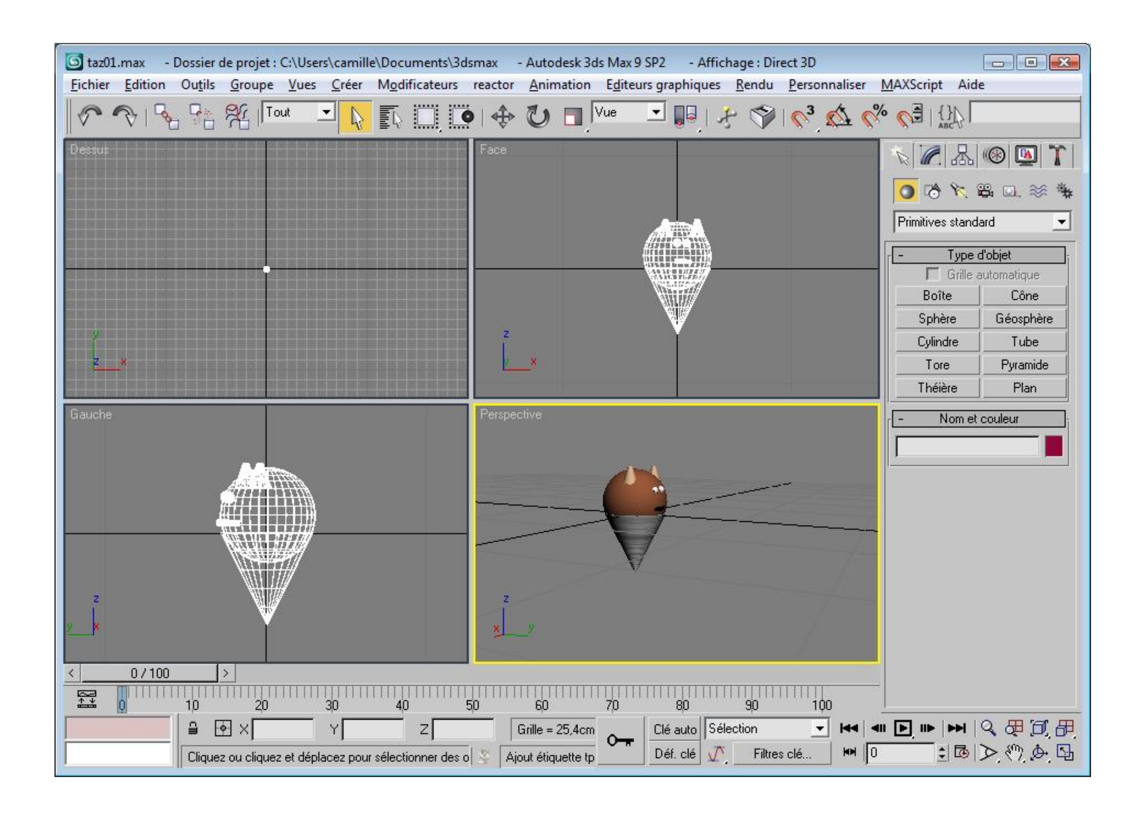
3.3.5 Les objets 3D avec 3D Studio Max 9 (Camille)

Aucun de nous ne sait utiliser 3DS Max... C'est pourquoi nous avons eu beaucoup de difficultés pour savoir comment nous allions modéliser Taz. Toutes nos recherches sur Internet pour trouver un modèle ont été vaines. Nous avons finalement décidé d'essayer de créer un personnage car nous ne voulions pas avoir un Schtroumpf dans notre jeu comme lors de la troisième soutenance. Nous avons choisi de faire une sphère au dessus d'un cône (le cône représentant la tornade de Taz lors de ses déplacements), avec des yeux, une bouche et des oreilles. Après de nombreux essais, nous avons aussi réussi à appliquer des textures à ce... personnage.

Malgré tout ces efforts, et toutes ces découvertes, nous avons rencontré un problème majeur. Non, ce problème n'est pas le grand manque de ressemblance entre notre personnage et Taz ! Nous ne savions (et ne savons toujours pas) comment récupérer l'ensemble des textures appliquées sur le modèle pour les appliquer dans le jeu. Nous avons essayé avec un peu d'aide extérieure, mais nous n'avons réussi qu'à récupérer une page avec les patrons des formes. Nous avons donc appliqué une texture unie au personnage dans le jeu.

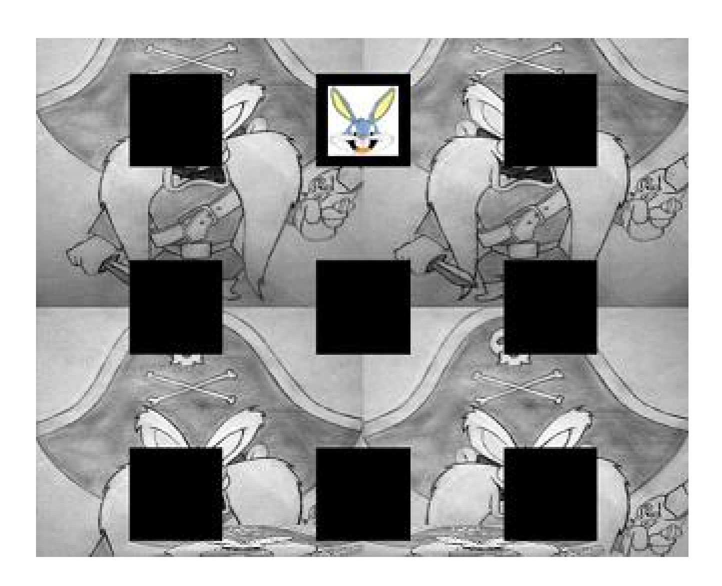
Nous avons ensuite modélisé quelques objets basiques pour ajouter aux décors du jeu (les pyramides du deuxième monde), mais nous avons toujours eu le même problème pour récupérer les textures appliquées sous 3DS Max.

Nous vous laissons maintenant admirer notre œuvre d'art...

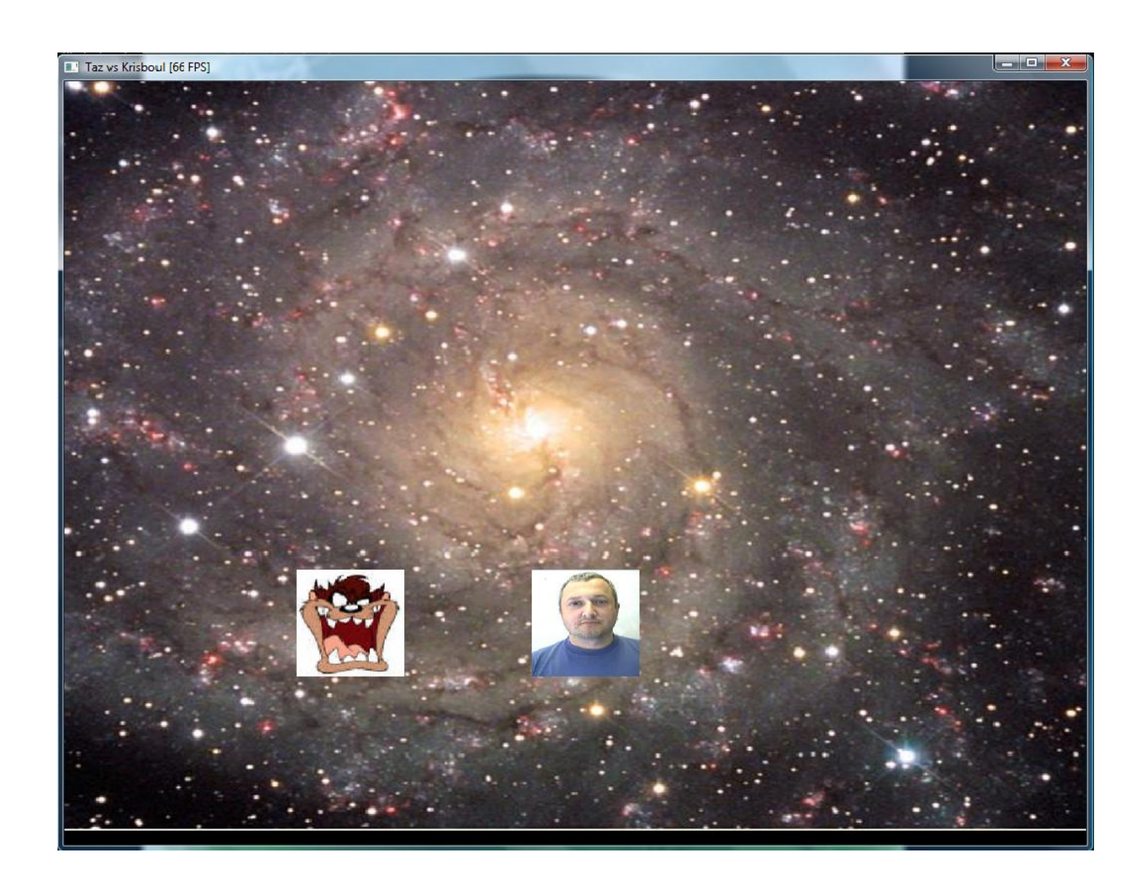


3.3.6 Les mini-jeux

A la fin de chaque niveau ont été crée des mini-jeux permettant le passage au monde suivant si celui-ci est gagné par le joueur. Le premier mini jeu est une chasse taupe mettant en scène bugs bunny et sam le pirate qui doit le taper. Si on le tape dix fois alors on gagne le mini-jeu et l'on peut passer au monde suivant. Il se présente de la manière suivante : on a neuf trous dans chacun apparait aléatoirement bugs que l'on doit frapper en appuyant sur l'une des neuf touches du clavier correspondantes qui sont respectivement de haut en bas « AZE QSD WXC ». Lorsque l'on touche bugs l'image change et un son de frappe se déclenche en même temps. Le mini-jeu est en 2D avec une caméra placée de façon a ne gérer que les coordonnées en x et y. Ensuite vient la création de la scène de jeu avec les trous, le décor ainsi que les images de bugs qui apparaissent aléatoirement ainsi que les bruitages.



Le deuxième mini-jeu n'est autre que le mini-jeu de fin : le combat entre Taz et Krisbool. Le but de ce mini-jeu étant de toucher trois fois krisbool pour gagner le mini-jeu (et le jeu par la même occasion). Ce mini-jeu est également en 2D et la scène est gérée de la même manière que le premier mini jeu sauf que là nous avons ajouté une gestion des mouvements de Taz et de Krisbool sachant que seul Taz peut sauter.



3.3.7 Les problèmes rencontrés

Les problèmes… je pense que c'est l'une de nos plus grandes parties parce que tout d'abord nous ne savions pas comment démarrer tout simplement car nous n'avions jamais codé auparavant ! Nous commencions un tout nouveau projet sans aucunes bases (ou presque)… Nous avons reçu de l'aide de la part de certains spé mais aussi grâce à de nombreux sites internet traitant d'OpenGL. Ensuite dans le moteur graphique en lui-même nous avons eus quelques problèmes pour que la caméra ne suive notre personnage que par rapport aux axes x et y car il suivait Taz également en lorsqu'il sautait. L'application des textures était quelque peu laborieuse à comprendre au début car on obtenait des textures mal appliquées ou trop étirées ce qui donnait un très mauvais rendu graphique.

Suite à la résolution de ces « petits » problèmes vint sur la table le loader 3DS… il est vraiment celui qui nous a posé le plus de problèmes. Je pense même que c'est la partie où l'on aura passé le plus de temps de tout notre projet ! Donc après avoir créé ce loader (et non sans mal !) nous avons décidé de le tester tout d'abord dans une fenêtre séparée… et là tout marchait correctement nous pensions enfin que le calvaire était fini… mais nous avions parlé trop vite… en essayant de l'implémenter dans notre jeu pour qu'il charge des modèles 3D il ne voulait tout d'abord pas le faire car il nous mettait plusieurs erreurs totalement inconnues… puis par la force des choses et après plusieurs heures de débugge intensif nous finîmes par trouver et notre modèle se chargeait bien mais on ne le voyait toujours pas ! Nous comprîmes peu de temps après qu'il était trop grand et qu'il fallait le redimensionner grâce à une fonction de glfw qui n'est autre que glScale. Lorsque le périple du loader 3D se termina, c'est un grand « OUF » de soulagement que nous émirent… La tâche fut rude mais nous l'avons finalement franchie…

3.4 Le moteur physique : Alain et Sébastien

3.4.1 Les prévisions

Le moteur physique est la partie du jeu dont nous pouvons être fiers. Dans nos prévisions, c'est ce qui a été le plus respecté dans l'avancé de notre projet. Nous l'avons fini pour la troisième soutenance et cela nous a permis de nous répartir, Alain et Sébastien, pour aider Camille et Hélène dans les autres domaines que nous n'avions pas finis. Le moteur physique est un élément mature et complet de notre jeu.

3.4.2 Les repères physiques

Pour pouvoir faire des comparaisons de coordonnées, nous avons tout d'abord entré toutes les coordonnées de nos objets physiques dans une... liste chaînée ! Grâce à ce type de données, nous pouvons faire des ajouts d'éléments en tête de liste très simplement avec des allocations mémoire et des pointeurs. Bien sûr, comme nous sommes très bons, nous libérons la mémoire proprement à chaque fin de monde.

Nous avons créé un nouveau type de données pour notre liste chaînée. Dans ces enregistrements nous avons : les coordonnées de début d'objet sur les axes X, Y et Z (six valeurs donc) et un pointeur sur le suivant, la fin de liste étant le pointeur nul (NIL en Delphi). Ces valeurs sont des constantes que nous définissions pour chaque monde.

3.4.3 Gestion des collisions

Nous avons une gestion assez simple, donc très puissante, des collisions. A chaque tour de notre boucle principale du jeu, on compare les coordonnées de notre personnage avec tous les objets de la carte, on fait un test t'intersection des coordonnées. Avec la liste chaînée et les pointeurs le calcul est très rapide et ne ralentit pas le jeu alors que nous pensions au début que ça bloquerait un peu le jeu. Comme notre moteur physique est très général et est aussi utilisable pour des objets qui se déplacent (oui, les « monstres »). Pour les monstres, nous avons par contre une autre liste chaînée avec des variables contrairement aux objets fixes.

3.4.4 La gravité

Notre projet ne serait pas réaliste sans un rendu complet de la réalité avec la gravité ! Nous avons en effet ajouté la gravité pour la deuxième soutenance. Taz peut se désormais déplacer de caisse en caisse pour échapper aux terribles monstres qui peuplent nos cartes. Un ajout de détection de saut (barre d'espace) et l'ajout des collisions sur l'axe Z (altitude) nous a permis d'avoir un rendu intéressant qui nous convient, toujours très léger à l'exécution car optimisé.

Concrètement, nous avons une variable booléenne qui se met à FALSE si on a déjà sauté et qu'on est en l'air. A chaque tour de boucle, tant que l'altitude n'est pas à 0 et qu'il n'y a pas de collision (donc pas sur une caisse), on fait descendre notre Taz d'une valeur constante. Une fois qu'on retombe sur le sol, notre variable booléenne se remet à TRUE et on peut resauter, de même si on est sur une caisse pour qu'on puisse se déplacer de caisse en caisse.

3.4.5 Les problèmes rencontrés

Nous avons surtout eu des problèmes d'organisation et d'ajustement par rapport aux autres domaines du projet. La mise en place d'un moteur physique qui puisse être utilisé en 2D et en 3D n'est pas simple. Notre répartition en petits groupes de deux a été plus que profitable. De cette manière, chacun de nous pouvait donner son avis pour pouvoir utiliser le moteur physique le mieux possible. C'est comme cela que nous avons pu définir un moteur physique à notre façon, qui ne contient rien d'inutile et qui correspond parfaitement à notre demande.

3.5 Le réseau : Camille et Sébastien

3.5.1 Les prévisions

Au début de l'année, nous étions totalement dans l'inconnu en ce qui concerne cette partie de notre projet. C'est pourquoi nous avons décidé de nous lancer dans la création d'un chat, ce qui nous permet de comprendre les bases de la création d'un réseau.

3.5.2 Le chat (pas celui de Junior)

Nous avons présenté notre réseau à la deuxième soutenance. Nous avions alors un chat qui fonctionnait correctement grâce aux modules TClientSocket et TServerSocket. Ce chat permet à deux personnes de communiquer. Il utilise le protocole TCP/IP pour envoyer et recevoir des messages tant en local que sur Internet. Cependant, nous n'avons pas été très convaincus par cette forme de réseau, c'est pourquoi nous avons préféré toucher à un autre aspect du réseau, en guise de bonus pour le jeu.

3.5.3 La mise en ligne des scores

Cette partie n'était initiallement pas prévue dans le cahier des charges. Nous avons créé un module d'envoi des scores dans Delphi. Le composant Indy a été indispensable pour réaliser cette partie. Après huit mois d'utilisation de Delphi, nous nous sommes aperçus que nous ne l'avions pas installé. La faute de Charles Vu qui nous avait dit de décocher cette case lors de l'installation. Ne sachant pas à quoi correpondait ce module, nous avons agit comme des moutons (bêêê) au début de l'année. Mais ce temps est révolu.

Une fois réinstallé Delphi, nous avons pu nous concentrer sur l'envoi des scores de manière plus concrète... On crée une classe qui regroupe les différents éléments à envoyer, en l'occurence le pseudo saisi par l'utilisateur ainsi que le score qu'il vient de faire. Pour un traitement simplifié des données sur le site Internet, nous effectuons une requête POST qui est traitée comme un formulaire en PHP.

3.5.4 Les problèmes rencontrés

Nous avons au un problème majeur : le chat ne fonctionnait que dans un sens ! C'est-à-dire qu'une seule personne pouvait écrire, et l'autre ne pouvait que lire. Après de nombreuses recherches nous n'avons pas réussi à résoudre ce problème, bien que nous connaissons la cause de ce problème... Il faudrait que les clients soient tous connectés à un serveur (mais comment faire ?) ou que chaque personne soit client et serveur en même temps.

3.6 L'Intelligence Artificielle : Camille et Hélène

3.6.1 Les prévisions

Nous avons décidé d'inclure une intelligence artificielle à notre jeu. Nous voulions créer des monstres qui compliqueraient la tâche de Taz. Le principe est donc de créer un périmètre autour des monstres, et si Taz entre dans ce périmètre, le monstre le suit.

3.6.2 Création des monstres et de leurs périmètres

Nous avons dessiné les monstres de la même façon que les obstacles et le personnage. Le problème a ensuite été de récupérer les monstres dans un tableau dynamique, nous avons finalement utilisé la même méthode que pour les obstacles : le tableau 'tabl_monst' contient les coordonnées maximales et minimales du monstre sur chaque axe. Nous avons aussi créé un enregistrement dans lequel il y a une variable 'périmètre' de type booléen. La variable 'périmètre' est à vrai si le personnage est entré dans la zone du monstre, à faux sinon.

L'étape suivante a été de déterminer le périmètre autour du monstre, c'est-à-dire la zone dans laquelle le personnage ne doit pas entrer s'il ne veut pas se faire attaquer. Pour cela, nous avons fait une fonction qui vérifie si les coordonnées du personnages sont comprises entre celles de la zone ou pas. La zone a alors été définie grâce aux coordonnées du monstre que l'on a récupérées dans le tableau 'tabl_monst' auxquelles on ajoute une variable, plus ou moins grande en fonction de la taille du périmètre que l'on désire. Si le personnage se trouve dans la zone, le monstre le suit.

3.6.3 Les problèmes rencontrés

Cette partie du jeu nous a posé beaucoup de problèmes. Tout d'abord, nous avons eu des difficultés à dessiner le monstre. Nous avons résolu le problème après plusieurs essais : il fallait jouer sur les glPushMatrix et glPopMatrix. De plus, pour animer notre monstre, nous avons utilisé la fonction gltranslatef, comme nous l'avions fait pour le déplacement du personnage. Cependant, les deux fonctions gltranslatef (celle qui anime le montre, et celle

qui permet au personnage de se déplacer) interfèrent. Nous avons alors trouvé plusieurs solutions, nous les avons testées, mais sans succès. D'abord nous avons essayé de dessiner le monstre dans une autre boucle de dessin, car il était jusqu'à maintenant dessiné dans la même boucle que le reste du jeu, mais cela n'a rien changé. Nous sommes donc revenus à notre boucle de dessin du départ, et avons essayé de mettre la fonction gltranslatef correspondant au déplacement du monstre à l'intérieur d'une autre fonction, pensant alors qu'il n'y aurait plus d'interférences avec la fonction gltranslatef du personnage.

Ce test n'a pas eu plus de succès que le précédent. Nous avons aussi tenté d'introduire le gltranslatef au sein de la fonction qui permet au monstre de se déplacer. Nous avons finalement réussi à résoudre ce problème, en dessinant le monstre ET en lui ordonnant de se déplacer dans une procédure vraiment séparée du reste. Ceci à marché, et nous avons donc réussi à avoir un monstre qui attaque Taz sans sortir de sa zone ! Nous avons cependant eu des difficultés tout au long de la création de l'IA qui étaient liées à ces trois fonctions (glPushMatrix, glPopMatrix et gltranlatef).

3.7 Le site Internet (Sébastien)

3.7.1 Présentation

Le site Internet a aussi été une grande aventure. Tout d'abord, m'intéressant aux logiciels libres, je me suis d'abord tourné vers le Système de Gestion de Contenu (en anglais CMS pour *Content Management System*) dénommé Joomla. Intéressé par sa modularité et sa possible personnalisation apparemment très simple, nous avons vite été découragé. En effet, nous avons un jeu modeste et nous souhaitons simplement poster des informations courtes ainsi que notre code source et il n'était pas nécessaire pour nous de disposer d'outils puissants (quoique compliqués finalement) pour cela. C'est pour cela qu'à la seconde soutenance, nous avons présenté une refonte totale de notre site, mais aussi plus légère. A titre d'exemple, à la première soutenance, le site faisait plus de 10 Mo, et pour la deuxième soutenance, il ne faisait que 9 Mo et nous avions mis en ligne nos codes sources...

3.7.2 Joomla ou xHTML avec des CSS à la main ?

Très clairement, une base de donnée poussée et une interface pour éditer nos articles en ligne n'ont pas fait le poids par rapport à un site fait main avec un simple éditeur de texte. Grâce au Site du Zéro que j'ai lu en une nuit (ça c'était cadeau), j'ai pu refaire un site fonctionnel et léger en xHTML, avec des CSS ainsi qu'un compteur en PHP, pouvant être facilement mis à jour grâce à des *include* bien placés. Nous avons désormais un site performant qui respecte les standards du Web, très important pour être sûr d'être bien lu par tous.

3.7.3 Les problèmes rencontrés

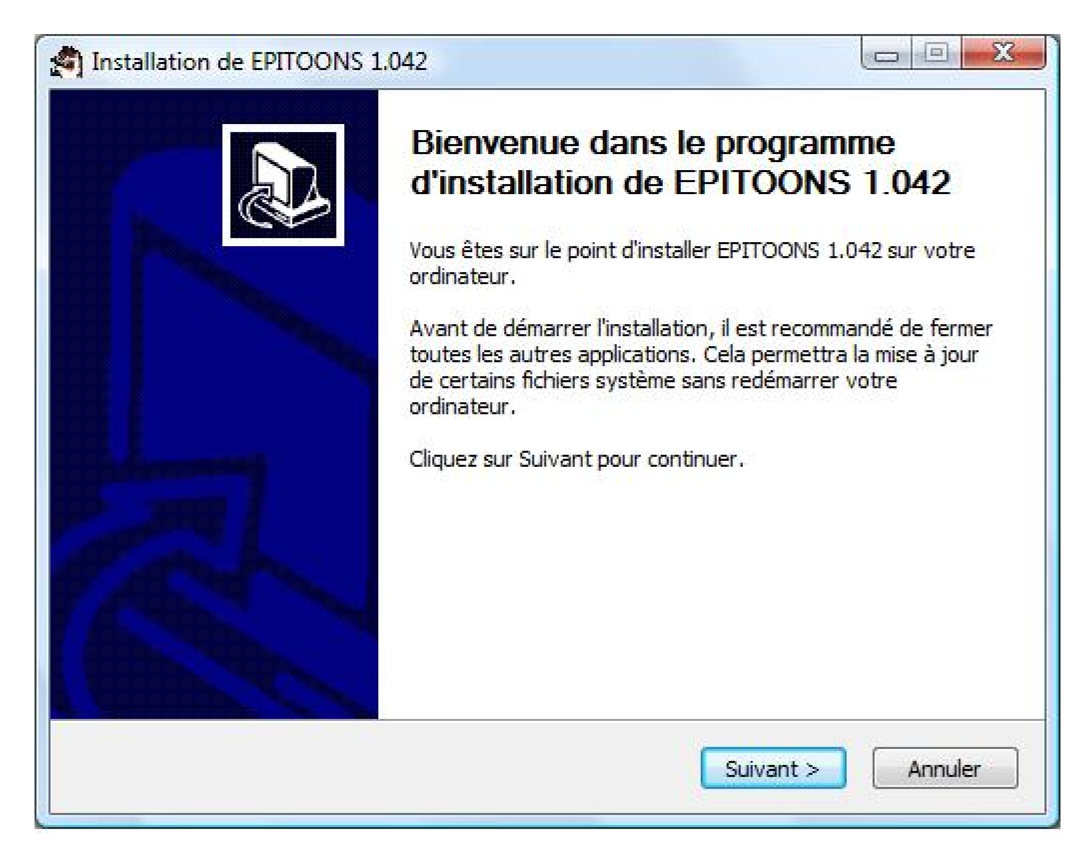
Plusieurs problèmes ont été rencontrés. Rien que pour avoir un site Web en local, comme le bocal n'est pas de notre côté le jour des soutenance, il a fallu trouver un moyen pour avoir un serveur Apache avec un support PHP sur Windows... Pour se faire, j'ai eu recours au logiciel libre WampServer qui signifie tout simplement Serveur Windows Appache MySQL PHP. Grâce à ce logiciel, nous pouvons avoir un site en local qui inclus des pages dynamiques.

Le plus gros problème que j'ai eu a été pour récupérer les scores du mini-jeu « Sam vs Bugs ». Comme je n'avais pas de connaissances en formulaires PHP, j'ai demandé au groupe Just-Pinagui de la promo 2011 de me donner un coup de main (pas trop fort) car ils avaient eu la même idée un an plus tôt. L'entraide a été immédiate et l'ancien chef de projet a vite répondu à ma demande. J'ai maintenant une page qui recueille les scores envoyés depuis le jeu et qui les affiches dans un tableau sur Internet (et en local également).

3.8 Finalisation

3.8.1 Installation, désinstallation

Pour l'installation, et la désinstallation par le même occasion, nous faisons appel à l'excellent logiciel libre NSIS (*NullSoft Scriptable Install System*). Le logiciel est sous forme de script que l'on compile (un peu le principe de LaTeX !). On y adjoint les fichiers d'installation de base ainsi que les librairies indispensables pour lancer le jeu, les manuels et le code source. Le logiciel permet en outre de configurer le répertoire d'installation des fichiers. On a le choix lors de l'installation d'installer tel ou tel composant. Un raccourcis permet à l'utilisateur de pouvoir désinstaller le logiciel et un autre d'avoir le lien hypertexte de notre site.



3.8.2 Jaquette

Je me suis occupé personnellement de la jaquette sous Photoshop en essayant d'obtenir un bon rendu graphique voir professionnel. Le résultat est assez satisfaisant pour ma part et a été approuvé par le reste de la team ! En effet on retrouve sur la face avant notre Logo de la TazManiaTeam (TMT) ainsi que les mentions légales notamment pour l'âge (3+). Au dos nous ne retrouvons qu'un screen du mini-jeu pour ne pas tout dévoiler mais également le texte racontant l'histoire du jeu. Nous avons ajouté la configuration conseillée pour jouer au jeu et enfin à nouveau des images légales (âge, code barre, avertissement pour l'épilepsie. . .)



Logo de la TaZmaniaTeaM



3.8.3 Les différents manuels

1. Manuel d'utilisation

   Pour faire ce manuel, nous nous sommes inspiré de manuels de jeux de différents types. Nous avons alors remarqué que certaines parties sont nécessaires dans tout manuel d'utilisation d'un jeu vidéo, tels que la prévention contre l'épilepsie, les aides pour le dépannage… Mais nous avons aussi voulu faire un manuel qui reflète l'esprit de notre jeu. Ce livret contient plusieurs parties :

   – **Prévention contre les risques d'épilepsie et règles à respecter pour jouer dans les meilleures conditions possibles** : pour respecter la norme de tous les manuels d'utilisation de jeu vidéo.

   – **Table des matières** : permet à l'utilisateur de trouver plus rapidement les informations qu'il cherche dans le livret.

- **Présentation du jeu** : peut permettre au joueur de comprendre l'histoire et de se mettre dans l'esprit du jeu, et par conséquent d'avoir une autre approche de celui-ci.

- **Toutes les commandes** : informe le joueur des touches à utiliser pour jouer.

- **Jouer au jeu Epitoons** : explique les différentes fonctions des boutons du menu du jeu.

- **Réussir un niveau** : explique le but à atteindre dans chaque niveau.

- **Réussir un mini-jeu** : décrit les mini-jeux et explique le but de chacun. Cette partie renseigne aussi le joueur sur la différence des mini-jeux pendant une partie complète ou en choisissant de ne jouer qu'à un mini-jeu seul.

- **Trucs et astuces** : conseille le joueur pour pouvoir gagner le jeu.

- **Dépannage** : permet au joueur de résoudre les différents problèmes qu'il peut rencontrer avec notre jeu.

- **Avertissement** : comme pour la première rubrique, celle-ci sert à respecter la norme de tous les manuels d'utilisation de jeu vidéo.

- **Nous contacter** : permet à l'utilisateur de nous poser directement ses questions en cas de problèmes.

2. Manuel d'installation

   Pour faire le manuel d'installation, je me suis dit que le plus simple était de se référer à des jeux déjà commercialisés. J'ai donc pris exemple sur des différents jeux pour que ce manuel soit conforme aux règles en vigueur. J'ai remarqué aussitôt que certaines parties composant ces manuels étaient indispensables, et devaient être incorporées dans notre manuel. Ces parties concernant le plus souvent le système d'exploitation. Ce livret contient plusieurs parties :

   – **Table des matières** : permet à l'utilisateur de trouver plus rapidement les informations qu'il cherche dans le livret.

   – **Configuration nécessaire** : Permet d'accéder aux différentes données de votre système telles que le processeur ou la carte graphique.

   – **Configuration minimale** : Contient toutes les caractéristiques que votre système doit posséder pour que EPITOONS fonctionne normalement.

   – **Nettoyage du système avant l'installation du jeu** : Contient les recommandations pour le nettoyage de votre PC et explique les différentes fonctions de Windows qui peuvent optimiser votre système.

   – **Utilisation de l'aide Windows** : Décrit le chemin afin d'accéder à l'aide Windows pour les différents systèmes.

   – **Installation du jeu** : Explique les actions à effectuer durant les différentes étapes permettant l'installion correcte du jeu EPITOONS.

   – **Désinstallation et réinstallation du jeu** : Explique le processus à accomplir pour désinstaller EPITOONS avec la possibilité de le réinstaller.

Pour plus de renseignements, lisez le manuel !

3.9 Les petits plus

3.9.1 Le menu du jeu (Sébastien et Camille)

Nous avons créé un menu de démarrage. En effet, ce menu nous permet de ne pas arriver directement sur le jeu (parce que c'était un peu moche...), mais aussi de faire un lien entre le jeu, le réseau, et les options qui arriveront plus tard. Nous avions tout d'abord utilisé la bibliothèque ShellAPI, et plus particulièrement la fonction ShellExecute qui nous permet d'ouvrir un fichier .exe du dossier. Cependant, nous nous sommes vite rendu compte qu'il y avait bien plus simple, et surtout plus efficace ! En effet, avec la fonction ShellExecute nous ne pouvions qu'ouvrir un .exe... ce qui posait problème pour ouvrir les sous menus car nous devions alors avoir un .exe pour chaque partie du jeu qui est liée au menu. Nous avons finalement juste appelé nos fonctions ou procédures lorsque l'utilisateur clique sur un bouton.

Le menu du jeu (menu central) donne accès au jeu directement par le bouton « Démarrer ». On peut aussi quitter le jeu grâce au bouton qui porte ce nom, ainsi que lancer le réseau. Ce menu donne aussi accès à trois sous menus : les options (voir partie suivante), un sous menu qui permet de joueur uniquement au mini jeu de son choix, et un autre qui permet de choisir le niveau à partir duquel on veut jouer.

3.9.2 Les options (Alain et Camille)

Nous avons créé un menu d'options qui nous permet de choisir la résolution de l'écran et la musique du jeu. Ce menu a été fait en TForm, comme le menu du jeu. Pour ce qui est de la résolution, l'utilisateur a quatre choix : 800 x 600, 1024 x 768, 1280 x 960 ou 1680 x 1050. Nous avons utilisé des variables qui contiennent la résolution choisie par le joueur. Ces variables sont ensuite utilisées tout au long du jeu pour que la résolution reste la même lors des passages aux mini jeux ou au monde suivant. Si le joueur ne fait pas de choix (c'est-à-dire s'il ne passe pas par le menu des options ou s'il clique sur « Annuler » quand il est dans ce menu), la résolution par défaut est 1024 x 768.

L'utilisateur peut aussi choisir de jouer en plein écran ou non. Pour jouer en plein écran il est obligé de passer par le menu des options. En effet, nous avons choisi de mettre le jeu en mode fenêtré par défaut. Nous avons utilisé pour cela le même principe que pour la résolution en utilisant des variables tout au long du jeu. Si l'utilisateur choisit le mode plein écran, on ouvre la fenêtre en utilisant GLFW_FULLSCREEN, sinon on utilise GLFW_WINDOW. Si le joueur choisit le mode plein écran, les mini jeux seront eux aussi en plein écran.

Enfin, les options permettent aussi au joueur de choisir la musique qu'il désire pour accompagner sa partie. La musique par défaut est celle des Looney Tunes. Dans le menu des options l'utilisateur a le choix entre trois autres musiques (celle des Looney Tunes étant par défaut, nous ne l'avons pas remise dans les choix des options), mais il peut aussi choisir de ne pas mettre de musique. La musique choisie sera en marche dans les deux niveaux, mais pas pendant les minis jeux.

Lorsque le joueur a choisi ses options, il peut choisir de joueur au jeu entier (en cliquant sur « Démarrer ») ou de ne faire qu'un niveau ou un mini jeu. Elles sont valables pour tous les cas !

3.9.3 L'Audio (Alain)

Le son est l'une des parties qui m'a posé le moins de problème dans sa mise en place dans le jeu. En effet j'ai utilisé une librairie très complète, Fmod, qui regroupe le nécessaire de fonctions pour charger un son quand on le désir. C'est ainsi que l'on peut charger un son prédéfini à partir des options qui se lancera directement lorsque l'on lancera le jeu. Si l'on venait à ne pas définir la musique alors une musique se chargerait automatiquement (celle que l'on aura mis par défaut).

Le son doit tout d'abord être initialisé pour que la librairie sache que l'on va charger une musique puis ensuite grâce à la fonction Fsound_Sample_load qui permet de charger le fichier mp3 souhaité et Fsound_Playsound qui permet de lire le fichier mp3 préalablement chargé. Quand on quitte le jeu on ferme la musique grâce à la fonction CloseSounds de Fmod en même temps que le jeu se termine.

3.9.4 Ajout du FPS (Sébastien et Alain)

Nous avons ajouté un compteur de FPS pour nous donner une idée de la légèreté de notre programme. Nous affichons ce compteur dans la barre de la fenêtre. Le principe est simple et très intuitif, on compte le nombre de boucle sur un intervalle de temps (500 ms pour être assez précis) que l'on multiplie par deux pour nous donner le nombre d'images par secondes. Au début, du fait de la synchronisation verticale de nos écrans et par le fait que nous ne chargions pas d'éléments en 3D, seulement des primitives, nous avions un compteur tout le temps égal à 60 FPS. Puis, pour la soutenance finale, nous avons ajouté des éléments en 3D et nous pouvons voir maintenant la lourdeur de ces éléments qui nous font souvent baisser le compteur à 40 voire 30 FPS dû à nos cartes graphiques pourtant puissantes.

4 Moyens

4.1 Les logiciels

Nous avons aussi utilisé différents logiciels :

- 3D Studio Max 9
- 7-Zip
- Adobe Reader
- bNetSoul
- Delphi 2006 et 2007
- Kate (sur Debian GNU/Linux)
- Microsoft PowerPoint
- Microsoft Windows Vista et XP PRO
- Microsoft Word
- MiKTeX
- Mozilla Firefox
- Mozilla Thunderbird
- MSN
- Nero
- Notepad++
- NSIS (NullSoft Scriptable Install System)
- OpenGL
- OpenOffice.org
- Paint
- Photoshop CS3
- TeXnicCenter
- WampServer
- WinSCP

4.2 Le budget

Matériel	Prix
Ordinateurs	4500 €
Carte Imangin'R	1200 €
OS, logiciels	0 €
Nourriture, boissons	10000 €
Dépenses supplémentaires	25000 €
Total	40700 €

4.3 Temps de développement

Le travail de groupe a été enrichissant pour tout le monde. En effet, aucun de nous n'avait la même manière de travailler avant. Nous nous sommes finalement organisé et avons trouvé notre rythme de travail. Même si nos avancées n'étaient pas toujours aussi rapides que nous l'avions prévu, ceci était dû aux difficultés rencontrées et non à l'entente du groupe.

Comme nous l'avions prévu lors de la répartition des tâches, chacun s'est intéressé à l'ensemble des parties du jeu, nous savions donc que nous pouvions compter sur les autres membres du groupe en cas de problème plus ou moins important. Nous avons tout de même respecté nos prévisions et le travail le plus important de chaque partie a été fait par les personnes prévus au début. Même si ce projet n'a pas toujours été facile à réaliser, nous avons réussi à respecter notre cahier des charges.

5 Conclusion

Au terme d'une année à EPITA, il est temps de voir le chemin que nous avons parcouru jusqu'ici. Ce projet n'a pas toujours été facile mais nous sommes assez fiers de ce que nous avons été capable de réaliser. En effet, nous sommes partis de zéro (voire même négatif) pour finalement arriver à une première création que nous pensons correcte. Le travail en autonomie n'est pas évident mais notre groupe est bien soudé. Cela nous a permis de nous surpasser et d'évoluer constamment. Nous avons pris du plaisir à concevoir ce jeu.

Nous vous remercions pour toute votre attention.

La TaZmaniaTeaM

6 Remerciements

Nous tenions à remercier personnellement pour leur aide :

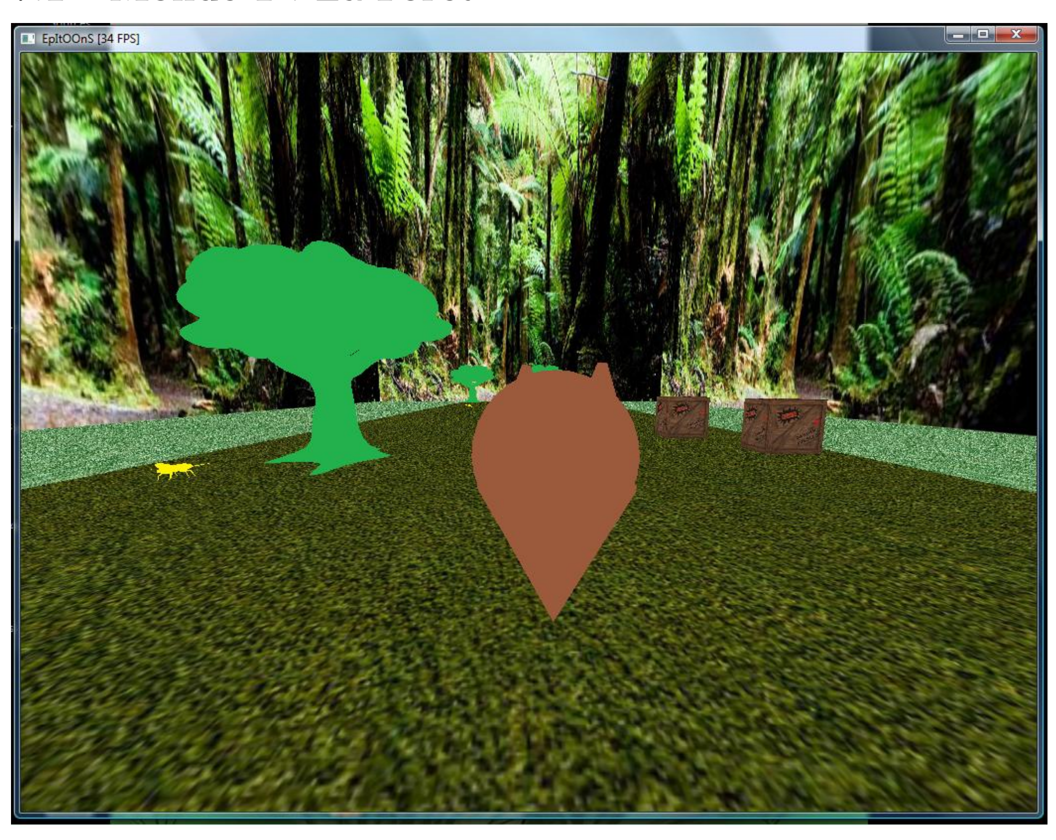
- Vincent NGUYEN (promo 2010) pour le loader 3D,
- Nicolas RINCK (promo 2011) pour le moteur graphique et physique,
- Pierre-Lou DOMINJON (promo 2011) pour l'envoi des scores sur Internet,
- Brice MANCONE (promo 2012) pour le moteur graphique,
- Aurélien MARTEL (promo 2012) pour le principe de l'Intelligence Artificielle,
- François PIETTE pour le composant ICS pour le réseau,
- Faouzi JAOUANI (promo 2011) pour le tutoriel OpenGL qui nous a permis de démarrer.

Et bien évidemment toutes nos familles et autres pour leur soutien :

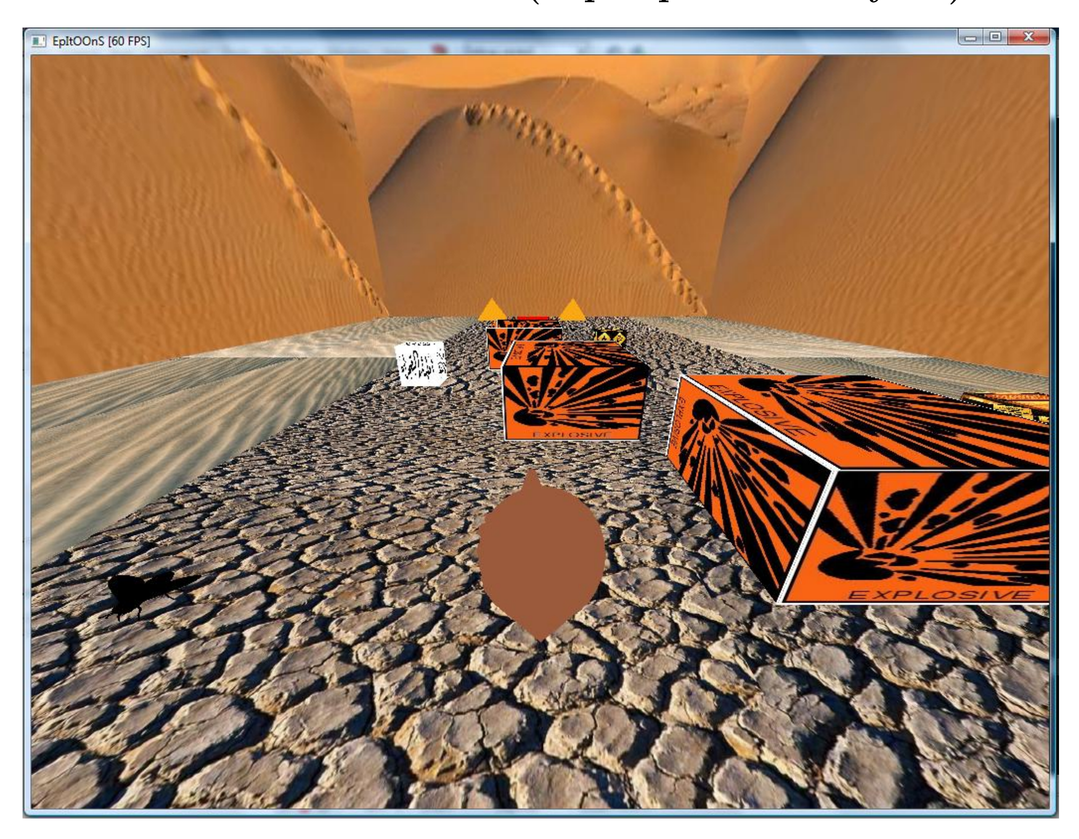
- Mathilde GUIRAUD pour avoir découpé les manuels et corrigé notre anglais,
- TAMA notre chat-mascotte (comme celui de Junior !),
- Ségolène LEPETIT pour toute son aide sur 3D Studio Max

7 Annexes

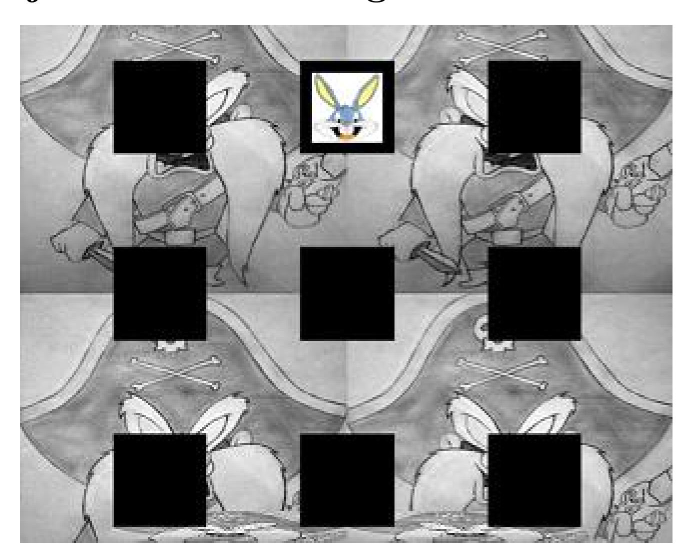
7.1 Monde 1 : La Forêt



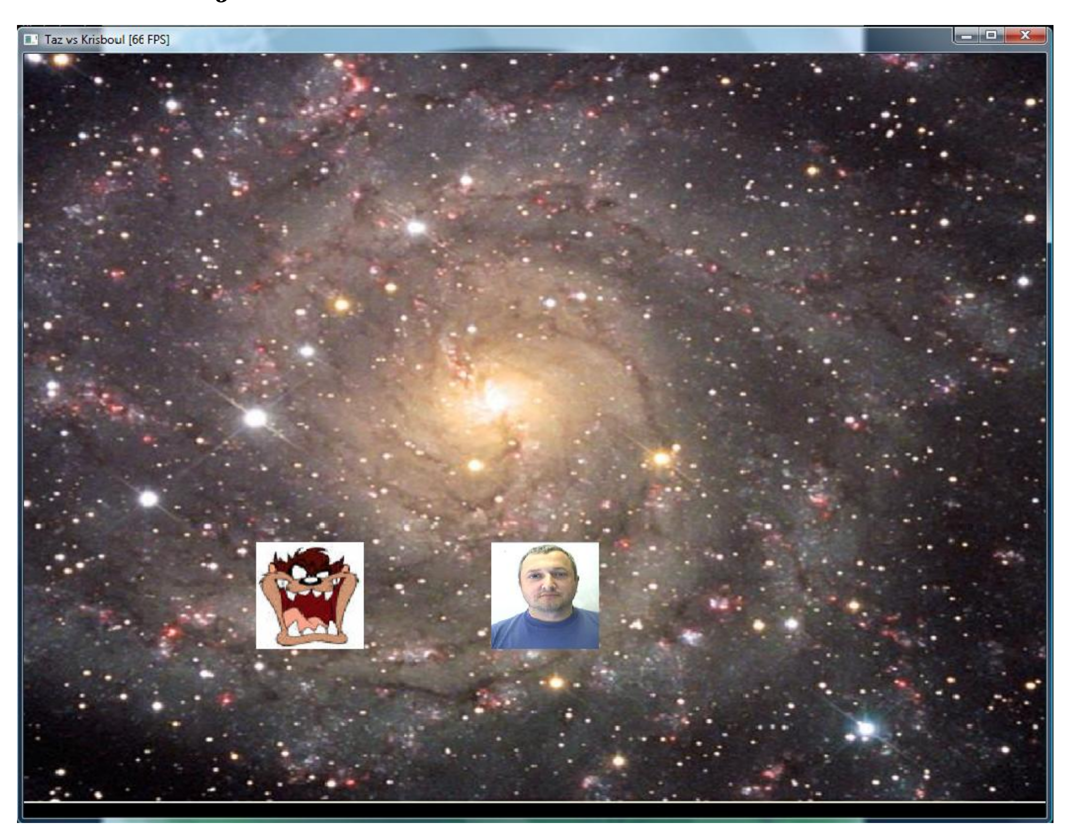
7.2 Monde 2 : Le Désert (Bip-bip et Vil Coyote)



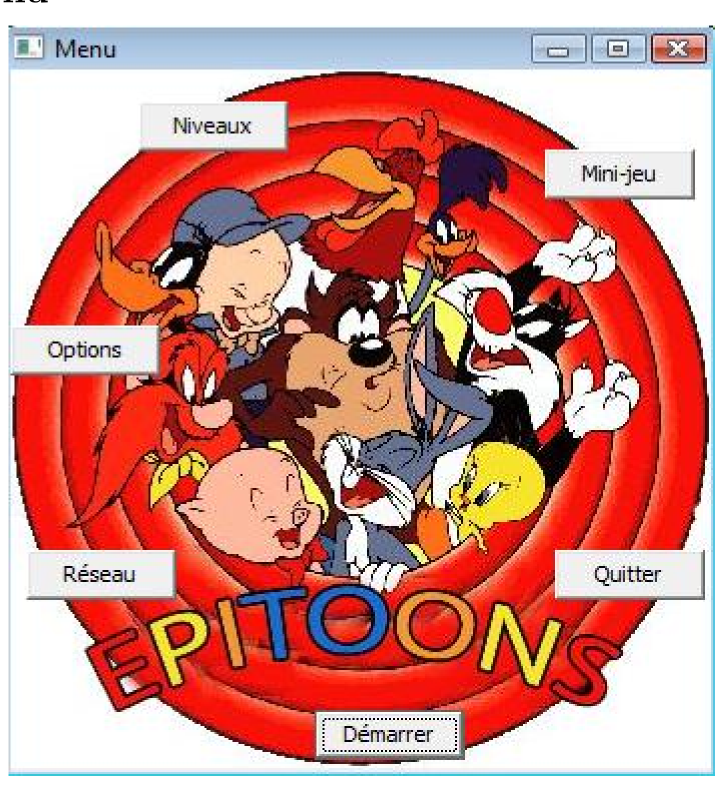
7.3 Mini-jeu 1 : Sam vs Bugs



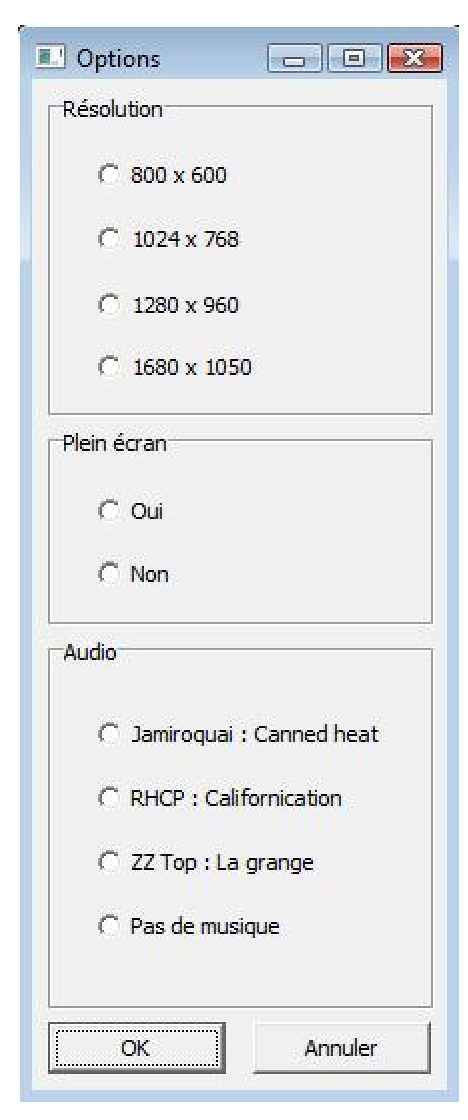
7.4 Mini-jeu 2 : Taz vs Krisboul



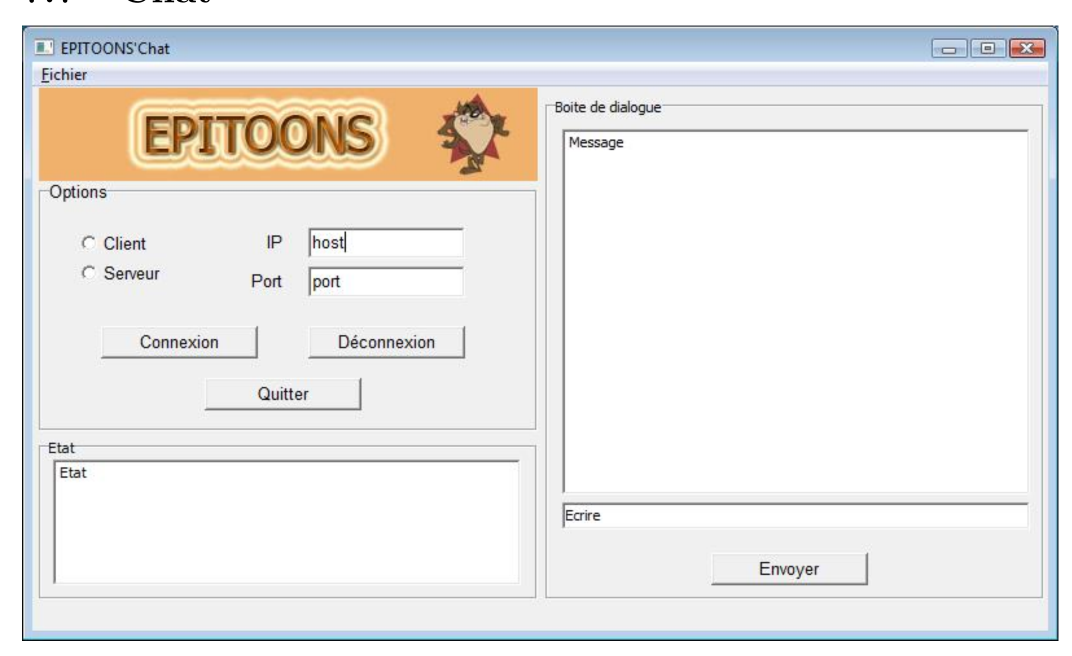
7.5 Menu


Menu
Niveaux
Mini-jeu
Options
Réseau
Quitter
EPITOONS
Démarrer

7.6 Les Options



7.7 Chat



7.8 Site Web

